Examen de février

Durée : 2 heures
Documents : autorisés

R. Rousseau
29 janvier 2003

Commencez par écrire votre nom et prénom sur la copie distribuée. Vous devez répondre aux questions dans les cadres prévus pour les réponses. Le système utilisé est Linux (ou Unix), le shell est `bash` (ou `zsh`) et l'invite est notée `%`. L'épreuve comprend quatre questions indépendantes avec le barème indiqué.

Question 1 : commandes de base *(5 points)*

Question 1.1 *(1 point)*

Ecrire une commande shell interactive (tapée au clavier) qui donne, aux fichiers au sens large situés sous le répertoire `~/unix/s04` et dont le suffixe est `.zsh`, les permissions de lecture et d'écriture au propriétaire (mais pas la permission d'exécution), la permission de lecture aux autres utilisateurs (mais ni la permission d'écriture, ni la permission d'exécution).

Question 1.2 *(2 points)*

Ecrire une commande shell interactive qui renomme les fichiers du répertoire courant (non récursif) dont le nom commence par une majuscule en leur donnant comme nouveau nom l'ancien nom avec un suffixe supplémentaire `.old`. Par exemple le fichier `Truc.c` sera renommé `Truc.c.old`. On donnera deux solutions, l'une avec une boucle `for` et l'autre avec une commande `find`.

Question 1.3 *(2 points)*

La commande `rpm -qa` liste sur *stdout* les noms de tous les logiciels installés (sur certains systèmes Linux), chaque nom sur une ligne. Par exemple :

```
SysVinit-2.83-5mdk
esound-0.2.29-1mdk
libglade0-0.17-6mdk
perl-GTK-0.7008-24mdk
...
```

- Donner une commande interactive qui compte les logiciels installés dont le nom ne se termine pas par `mdk`.
- Donner une commande interactive qui affiche les noms des logiciels installés sans la mention de version (cette mention est la partie située à la fin du nom, après le premier tiret, par exemple `0.17-6mdk` est la mention de version du logiciel `libglade0`). On utilisera la commande `sed` pour supprimer cette mention.

Question 2 : sed *(4 points)*

On considère un fichier de texte `allo.txt` qui donne pour chaque employé d'une entreprise une ligne de champs, séparés par un caractère ':'. Ces champs correspondent au prénom, nom, numéro de télephone et email des employés :

```
Paul:Dupont:0144567789:dupont@club.fr
Julie:Dalbanot:0493456789:julie@doo.fr
Henri:Pouliet:0493262645:pouliet@loa.com
...
```

Question 2.1 *(2 points)*

Donner un script `sed` qui transforme le fichier `allo.txt` en permutant le nom et le prénom, et en remplaçant le séparateur ':' par une barre verticale '|'.

Question 2.2 *(2 points)*

Donner un script `sed` qui transforme le fichier `allo.txt` en présentant les champs de chaque employé sur 3 lignes;

```
Paul Dupont :
   tél:   0144567789
   email: dupont@club.fr
...
```

Question 3 : awk *(3 points)*

On considère un fichier de texte `pib95.txt` contenant les PIB en millions de dollars US et la population des différents pays du monde, en 1995. Les champs sont séparés par un caractère de tabulation :

```
PAYS POP.95 PIB95
ALBANIE 3500000 2410
ETATS-UNIS 263000000 7476750
FRANCE 58100000 1463550
MOZAMBIQUE 17400000 1490
SUISSE 7000000 273340
...
```

Ecrire un script `pib95.awk` qui affiche :

- le nombre d'habitants total de la terre,
- le pays qui a le PIB/habitant le moins élevé,
- la moyenne des PIB/habitant de tous les pays (en dollars US).

On n'oubliera pas de sauter la première ligne d'entêtes pour ces calculs.

Question 4 : scripts shell et awk *(8 points)*

On considère le script Shell `analyse` suivant, qui ne fait qu'analyser ses arguments d'appels positionnels et affecter deux variables `OPT` et `CIBLE`

```
#! /bin/bash
USAGE="syntaxe: allo [-t|-e] nom|prénom"
if (( $# == 2 ))
then
    case $1 in
        -t ) OPT=3 ;;
        -e ) OPT=4 ;;
        * )  ### E1
             echo "err1: $USAGE" >&2 ; exit 1 ;;
    esac
    if [[ $2 == -* ]]
    then
        echo "err2: $USAGE" >&2 ; exit 2
    else  ### E2
        CIBLE=$2
    fi
elif (( $# == 1 )) && [[ $1 != -* ]]
then  ### E3
    OPT=3
    CIBLE=$1
else
    echo "err3: $USAGE" >&2 ; exit 3
fi
echo "OK: OPT=$OPT CIBLE=$CIBLE"
```

Question 4.1 *(1 point)*

Lorsque qu'on passe au point marqué ### `E1` dans le script `analyse`, combien d'arguments ont été transmis et que sait-on de leur valeur ?

Question 4.2 *(1 point)*

Lorsque qu'on passe au point marqué ### `E2` dans le script `analyse`, combien d'arguments ont été transmis et que sait-on de leur valeur ?

Question 4.3 *(1 point)*

Lorsque qu'on passe au point marqué ### `E3` dans le script `analyse`, combien d'arguments ont été transmis et que sait-on de leur valeur ?

Question 4.4 *(2 points)*

Lorsqu'on effectue les appels suivants au script `analyse`, qu'est-ce qui apparait à l'écran ?

```
analyse -a
analyse -t 'Julie'
```

Question 4.5 *(3 points)*

On utilise le fichier de texte `allo.txt` de la question 2. On construit un script Shell `allo`, dont le mode d'emploi est le suivant :

```
syntaxe:     allo [-t|-e] nom|prénom
sémantique:  affiche sur la voie stdout le prénom, nom et,
             soit le numéro de téléphone (par défaut ou option -t),
             soit l'email (option -e) de tous les employés spécifiés par une expression
             régulière de base qui sélectionne la totalité de leur prénom ou de leur nom.
exemples:
% allo -t Dupont
Paul Dupont 0144567789
% allo -e 'Dal.*'
Julie Dalbanot julie@doo.fr
% allo '.*uli.*'
Julie Dalbanot 0493456789
Henri Pouliet 0493262645
...
```

En appelant le script `analyse` pour analyser les arguments du script `allo`, donner le code du script Shell `allo`. On utilisera dans ce script la commande `awk` avec un miniscript en ligne de la forme

```
awk -F: "mini-script-awk" allo.txt
```

L'option -F précise le séparateur de champs. On prendra garde aux variables citées dans le miniscript avec le signe $, selon qu'il s'agit de variables du Shell (à évaluer immédiatement) ou de variables Awk (à protéger). Vous pouvez aussi considérer que le script `analyse` est une fonction interne du script `allo`, mais dans ce cas, le préciser.